Software Communications Architecture Specification

APPENDIX A. GLOSSARY

Revision Summary

1.0	Initial Release
1.1	no changes
2.0	no changes
2.1	no changes
2.2	no changes
2.2.1	no changes
3.0	No changes.

APPENDIX A GLOSSARY

A.1 ABBREVIATIONS AND ACRONYMS.

Abbreviation	Definition
802.x	IEEE network interface standards
AEP	Application Environment Profile
API	Application Program Interface
ASIC	Application Specific Integrated Circuit
BIT	Built-In Test
BSD	Berkeley Software Distribution
BSP	Burst Schedule Packets
BTS	Base Transceiver Station
C++	a computer programming language
C^4I	Command, Control, Communications, Computers and Intelligence
CF	Core Framework
COMSEC	Communication Security
CORBA	Common Object Request Broker Architecture
COTS	Commercial Off the Shelf
CPU	Central Processing Unit
DCD	Device Configuration Descriptor
DMD	*DomainManager* Configuration Descriptor
DoD	Department of Defense
DPD	Device Package Descriptor
DSP	Digital Signal Processor
DTD	Document Type Definition
FPGA	Field Programmable Gate Array
GIOP	General Inter-ORB Protocol
GPP	General Purpose Processor
GPS	Global Positioning System
HCI	Human-Computer Interface
HF ALE	High Frequency – Automatic Link Establishment
HH	Hours
HQ	Have Quick, an electronic counter-countermeasures waveform

Abbreviation	Definition
HW	Hardware
I/O	Input/Output
ICD	Interface Control Document
ID	Identification, Identifier
IDL	Interface Definition Language
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronic Engineers
IIOP	Internet Inter-ORB Protocol
INFOSEC	Information Security
I/O	input/output
IOR	Interoperable Object Reference
IP	Internet Protocol
ISO	International Standards Organization
Java	Computer Programming Language
JPO	Joint Program Office
JTA	Joint Technical Architecture
JTR	Joint Tactical Radio
JTRS	Joint Tactical Radio System
LAPx	Link Access Protocol x (where x represents 1 of several protocols defined by industry
MAC	Medium Access Control, a sublayer of the OSI Data Link Layer
MIB	Management Information Base
MLS	Multi-Level Security
MM	Minutes
MSB	Most Significant Bit
MSRC	Modular Software-Programmable Radio Consortium
MISSI	Multilevel Information System Security Initiative
N/A	Not Applicable
NAPI	Networking Application Programming Interface
NSA	National Security Agency
OE	Operating Environment
OMG	Object Management Group
OO	Object Oriented

Abbreviation	Definition
ORB	Object Request Broker
ORD	Operational Requirements Document
OS	Operating System
OSD	Operational Security Doctrine
OSI	Open System Interconnection
OTAR	Over-the-air Rekey
PCI	Peripheral Component Interconnect (bus)
PMCS	Programmable Modular Communication System
PN	Pseudo random Noise
POSIX®	Portable Operating System Interface
PPP	Point-to-Point Protocol
PSE52	Real-time Controller System Profile, defined in IEEE Std 1003.13
QoS	Quality of Service
RAM	Random Access Memory
RF	Radio Frequency
RS-232	Electronic Industries Alliance interface standard
RS-422	Electronic Industries Alliance interface standard
RS-423	Electronic Industries Alliance interface standard
RS-485	Electronic Industries Alliance interface standard
SA	Situation Awareness
SAD	Software Assembly Descriptor
SCA	Software Communications Architecture
SCD	Software Component Descriptor
SDD	Service Definition Description
SINCGARS	Single Channel Ground/Airborne Radio System
SLIP	Serial Line Internet Protocol
SNMP	Simple Network Management Protocol
SPD	Software Package Descriptor
SRD	Support and Rationale Document (for the SCA)
SW	Software
TBD	To Be Determined
TBR	To Be Reviewed

® POSIX is a registered trademark of the Institute of Electrical and Electronics Engineers, Inc.

Abbreviation	Definition
TCP	Transmission Control Protocol
TOD	Time Of Day
TRANSEC	Transmission Security
UML	Unified Modeling Language
UNIX	A computer operating system developed by AT&T Bell Laboratories.
UUID	Universally Unique Identifier
VME	VersaModule Eurocard, a 32 bit data bus standard
XML	eXtensible Markup Language

A.2 DEFINITIONS.

Aggregated Device

An aggregated Device is one that is part of a composite Device.

Application

The SCA defines an *Application* interface class that provides the interface for the control, configuration, and status of an instantiated application. An *Application* controls its components and establishes connections to other applications.

application

An executable software program which may contain one or more modules. The executable software exhibits pre-determined functionality (e.g. a waveform).

ApplicationFactory

An *ApplicationFactory* can be used to create an instance of an Application. The *DomainManager* creates an *ApplicationFactory* for each Software Assembly Descriptor that is installed.

Application Program Interface

An Application Program Interface (API) is the definition of operations and attributes contained in a set of related interfaces that provide a coherent functional capability

assemblycontroller

The *assemblycontroller* element indicates the component that is the main resource controller for the assembly.

Attribute (IDL)

An IDL attribute is a variable that contains a value of a specific type. Attributes may be declared with read-write or read-only access, and the appropriate get and set operations are generated when the IDL is compiled.

Attribute (SCA)

An SCA attribute is a property that describes a level of capability available in a *Device* or required by a *Resource*.

Client

A component that invokes an operation of another component.

Commercial Standard

A Commercial Standard is a set of requirements maintained for common use by industry. As used in this specification, commercial standards are available for use without restrictive licensing and are supported by commercially available hardware or software.

Component

A software module or element that implements an integral set of functionality. A component is described by a Software Package Descriptor.

Composite *Device*

A composite *Device* uses the CF *AggregateDevice* interface and is composed of one or more aggregated *Devices*. The composite *Device* and its aggregated *Devices* are strongly associated and have the same lifetime (i.e. removal of the composite *Device* removes the aggregated *Devices*).

Consumer

A software component that can receive user data traffic.

CORBA Component

A software component that implements CORBA interfaces (facets). A CORBA component is described by a Software Component Descriptor. A CORBA component encapsulates its internal representation and implementation; it provides a well-defined interface (operations, attributes, exceptions, and events) to component clients via CORBA IDL.

Core Application

An application program implementing the CF Framework Control Interfaces (e.g. *DomainManager*, and *ApplicationFactory*).

Core Framework

The Core Framework is the essential ("core") set of open application-layer interfaces and services to provide an abstraction of the underlying software and hardware layers for software application designers.

Destroy

The act of releasing / terminating a software object.

Device

1. Hardware device refers to a physical hardware element (typically a module performing a function or set of functions identified in its Device Profile).
2. The SCA defines a *Device* interface class. This logical *Device* interface is an abstraction of a hardware device that defines the capabilities, attributes, and interfaces for that device.

Device Configuration Description (DCD)

A Device Configuration Descriptor is an element of the Domain Profile that contains information about the children CF *Devices* for a CF *Device*, how to find the CF *DomainManager*, and the configuration information (Log, CF *FileSystems*, etc.) for a CF *Device*.

Device Driver

The low-level software, at the physical layer, that controls the physical interface a device uses for communication, e.g. to a hardware bus.

Device Package Descriptor (DPD)

A Device Package Descriptor is an element of the Domain Profile that contains information usable by system operators and maintainers, accessed via the HCI. It has properties that define specific information (manufacturer, model number, serial number, etc.) about the device.

Device Profile

Device Profile is the description of a hardware devices and may be made up of multiple Domain Profile files (Device Package Descriptor, Device Configuration Descriptor, Properties File). Device Profiles are part of a logical Device's Software Profile. This XML profile defines the kind of hardware device and the applicable attributes for type of device. The Device Profile also indicates if the device has load and execute behavior and the attributes that can be queried and configured

Document Type Definition (DTD)

A document type definition, or DTD, contains markup declarations that provide a grammar for a class of documents.

Domain

A Domain defines a set of hardware devices and available applications under the control of a single *DomainManager* component.

DomainManager

A *DomainManager* manages a set of available hardware devices and Applications. It is responsible for the set-up and shut-down of *Applications*, for allocating *Resources*, *Devices*, and non-CORBA components to hardware devices.

Domain Profile

The hardware devices and software components that make up an SCA system domain are described by a set of files that are collectively referred to as a Domain Profile. The *DomainManager* uses the Domain Profile to build its internal information base from the descriptions of the individual hardware devices, software components, and application assemblies under its control.

Event Service

The Event Service is a CORBA service that decouples the communication between components. The CORBA Event Service defines two roles for components: the producer role (produce event data) and the consumer role (process event data). Event data are communicated between producers and consumers by issuing standard CORBA requests.

Event Channel

An Event Channel is an intervening component that allows multiple producers to communicate with multiple consumers asynchronously. An event channel is both a consumer and a supplier of events. Event Channel is the intermediary between the components (producers) being changed and components (consumers) interested in knowing about changes. Event Channels that provide change notification can be general purpose, well-known components (Incoming and Outgoing Domain Management Event Channels) that are run as part of a domain-wide framework or specific-to-task components (e.g., temporary Event Channels that are created at application deployment).

Host

A host is a computer/processor and/or software application that provides services to one or more elements connected to it. These services may include, but are not limited to, network access, program loading, database storage, and HMI. The element or elements connected to a host may be

hardware elements (e.g. FPGAs), processing elements (e.g. DSPs), or a combination of elements (e.g. a JTRS radio).

Incoming Domain Management Event Channel

Incoming Domain Management Event Channel is an event channel that is internal to the domain and is used by domain's components (e.g., Devices) to send events to domain management components (*Application*, *ApplicationFactory*, *DomainManager*).

Initialize

The operation of setting a component to a known initial state.

Name

A user-friendly label such as the name used in DTDs of the Domain Profile.

Outgoing Domain Management Event Channel

Outgoing Domain Management Event Channel is an event channel that is external to the domain and is used by external domain's components (e.g., HCI) to receive events by domain management components (*Application*, *ApplicationFactory*, *DomainManager*).

Port

A port identifies a source /consumer (Provides Port) or a sink /producer (Uses Port) for data and/or commands. A *Port* specifies the types of data and commands accepted.

Primitive

An abstract, implementation-independent representation of the interactions between service users and service providers.

Private

As used in the SCA, a proprietary interface definition.

Producer

A software component that can supply user data traffic.

Profile Descriptor

A Profile Descriptor is an element of the Domain Manager that contains an absolute file name for either a Software Package Descriptor, Software Assembly Descriptor, or a Device Configuration Descriptor.

Properties Descriptor

A Properties Descriptor is an element of the Domain Profile that contains information about the properties applicable to a software package or a device package such as configuration, test, execute, and allocation types.

Property

An SCA Property is a variable that contains a value of a specific type. Configuration Properties are parameters to the Configure and Query operations. Allocation Properties define the capability levels required of a *Device* by a *Resource*.

Provides Ports

A *Port* that implements a specific interface.

Public

As used in the SCA, an open, publicly defined, non-license bearing interface definition.

Push Model

A Push Model is a mechanism where producers "push" event data to consumers; that is, producers communicate event data by invoking push operations on a "Push" interface (e.g., CosEventComm PushConsumer).

Real-Time CORBA

Real-Time CORBA is an emerging specification that provides additional interfaces to support typical real-time controls such as priorities, thread pools, and guaranteed message transfer times. Real-Time CORBA does not guarantee necessarily high-performance speed of operations.

Release (from the CORBA Environment)

When a CORBA object is released, it is no longer able to process object requests; its CORBA object reference unavailable to other objects. A release is analogous to the POA concept of deactivation. When a server object is deactivated, the association between the CORBA object and its implementation is removed. In the SCA, a component is removed from the OE and OE resources consumed by a component are returned back to the OE. For CORBA components, this includes deactivation. After a component is removed from the OE, a client is unable to communicate with the component.

Resource

A software component that implements the SCA defined *Resource* base application interface. The *Resource* interface provides an API for interaction between a *Resource* and the Core Framework and for interconnection of *Resources*. All visible SCA-conformant components of a user application must implement the *Resource* interface. Note that a logical *Device* is derived from *Resource* and must also implement the *Resource* interface.

Service Applications

Service applications are software programs running in the system that provide functionality available for use by other applications.

Software Assembly Descriptor (SAD)

A Software Assembly Descriptor is an element of the Domain Profile that contains information about the components that make up an application.

Software Component Descriptor (SCD)

A Software Component Descriptor is an element of the Domain Profile that contains information about a specific SCA software component (CF *Resource,* CF *ResourceFactory,* CF *Device*).

Software Package Descriptor (SPD)

A Software Package Descriptor is an element of the Domain Profile that identifies a software component implementation(s). General information about a software package, such as the name,

author, property file, and implementation code information and hardware and/or software dependencies are contained in a Software Package Descriptor file.

Software Profile

A Software Profile is a part of the Domain Profile and consists of either a Software Assembly Descriptor (for applications) or a Software Package Descriptor (for all other software components and hardware devices).

Standard Event

Standard Events are data constructs that describe an occurrence that has taken place within a JTRS. Standard Events will be produced and consumed using the Event Service. The producers and consumers of Standard Events are Core Framework (CF) entities, non-CF System entities, and waveform applications. The Standard Events will not be used for hard real-time control, however they will be used to allow the radio to take appropriate action as a result of the occurrence of an event.

Thread

A thread of execution control within an address space shared by other threads. Threads may receive events and pass messages to other threads, but do not have a protected address space. Threads are more efficient than processes (lightweight).

Unique Identifier

An identifier, guaranteed to be unique.

Uses Port

A port that implements the CF *Port* interface and uses 1 or more interfaces.

Waveform

A Waveform is the set of transformations applied to information that is transmitted over the air and the corresponding set of transformations to convert received signals back to their information content.